**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**
**BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES**

| | | |
|---|---|---|
| In re Application of: | ) | |
| | ) | |
| Inventor: Craig Storms et al. | ) | Examiner: Jacob F. Betit |
| | ) | |
| Serial No.: 10/038,071 | ) | Group Art Unit: 2164 |
| | ) | |
| Filed: January 4, 2002 | ) | Appeal No.: _________ |
| | ) | |
| Title: LIGHTWEIGHT SELF-CONTAINED | ) | |
| SELF-EXPANDING PRODUCT DATA | ) | |
| PACKAGE | ) | |

# REPLY BRIEF OF APPELLANTS

MAIL STOP APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In accordance with 37 CFR §41.41 and 41.37, Appellants hereby submit the Appellants' Reply Brief on Appeal from the final rejection in the above-identified application, as set forth in the Office Action dated February 4, 2008 and the Examiner's Answer dated October 6, 2008.

No fees are due at this time. Please charge any additional fees or credit any overpayments to Deposit Account No. 50-0494 of Gates & Cooper LLP.

## I. REAL PARTY IN INTEREST

The real party in interest is Autodesk, Inc., the assignee of the present application.

## II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences for the above-referenced patent application.

## III. STATUS OF CLAIMS

Claims 1-89 remain pending in the application.

Claims 1-11, 25-37, 51-63, and 77-89 have been withdrawn from consideration.

Claims 12-24, 38-50, and 64-76 stand rejected.

The rejections of claims 12-24, 38-50, and 64-76 are being appealed.

## IV. STATUS OF AMENDMENTS

No amendments to the claims have been made subsequent to the Office Action.

## V. SUMMARY OF CLAIMED SUBJECT MATTER

Independent claims 12, 38, and 64 are generally directed to the generation of a self-expanding data package for product data (see paragraph [0006]-page 3, lines 14-20, FIG. 7). Specifically, values in a set of constant lists are generated and stored in the data package (see paragraphs [0030]-[0032]- page 8, line 20-page 9, line 14, [0036]-page 10, lines 11-15, and [0061]-page 21, lines 2-8, FIG. 7 - 700 and 702). In addition, calculations (that operate on the values) are generated and stored in the data package (see paragraphs [0030]-[0032]- page 8, line 20-page 9, line 14, [0037]-page 10, line 18-page 11, line 4, and [0061]-page 21, lines 2-8, FIG. 7 - 704).

Once the values and calculations are stored in the data package, the data package is transmitted to a second computer system that expands the data package (see paragraphs [0065]-page 21, line 24-page 22, line 5 and [0067]-page 22, lines 16-20). The claim limitations provide that the package is expanded into a table having rows (see paragraph [0067]-page 22, lines 16-20, FIG. 7 - 708). In addition, the expansion is performed by combining each value with other parameters (i.e., in the data package, FIG. 7 - 708) and performing the calculations (from the data package) on the values (see paragraph [0067]-page 22, lines 16-20; FIG. 7 - 708). As claimed, each expanded row of the table represents a product and comprises one of the combinations (see paragraphs [0006]-page 3, lines 14-20, [0007]-page 3, line 21-page 4, line 3, [0026]-page 7, line 22-page 8, line 4, [0057]-page 19, line 23-page 20, line 4, [0067]-page 22, lines 16-20, and [0068]-page 22, line 21-page 23, line 2; FIG. 7 - 708). Further, the calculations eliminate one or more rows from the table (see paragraphs [0007]-page 3, line 21-page 4, line 3, [0064]-page 21, lines 19-22, [0066]-page 22, lines 6-15, and [0068]-page 22, line 21-page 23, line 2; FIG. 7 - 708). Accordingly, all of the information for expanding the data package is contained within the data package itself. In other words, the values for the set of constant lists and the calculations performed on the values are both generated and then stored in the data package.

In addition, Applicants note that the claims explicitly provide for and recite product data and that each row represents a product (see paragraphs [0026]-page 7, line 22-page 8, line 4, [0057]-page 19, line 23-page 20, line 4, [0067]-page 22, lines 16-20, and [0068]-page 22, line 21-page 23, line 2; FIG. 7 - 708). Accordingly, the invention is specific to product data. The invention solves a specific problem with publishing and consuming product data by packaging up all part properties for a related family of parts in a compact form, with a mechanism defined for assembling the individual properties into normalized tables. The invention affects both the publishers and consumers of this part data in a way that they must understand.

The charts below provide support for each claim limitation.

| CLAIM LIMITATION | SPECIFICATION/DRAWING SUPPORT |
|---|---|
| 12. A method for generating product data in a self-expanding data package in a computer system comprising: | [0006]-page 3, lines 14-20, FIG. 7 |
| generating one or more values in a set of one or more constant lists and storing said one or more values in the self-expanding data package, wherein the self-expanding data package is for product data; | [0030]-[0032]- page 8, line 20-page 9, line 14, [0036]-page 10, lines 11-15, and [0061]-page 21, lines 2-8, FIG. 7 - 700, 702, and 708; [0026]-page 7, line 22-page 8, line 4, [0057]-page 19, line 23-page 20, line 4, [0067]-page 22, lines 16-20, and [0068]-page 22, line 21-page 23, line 2 |
| generating one or more calculations that operate on one or more values in the set of one | [0030]-[0032]- page 8, line 20-page 9, line 14, [0037]-page 10, line 18-page 11, line 4, and |

|  |  |
|---|---|
| or more constant lists and storing said one or more calculations in the self-expanding data package; | [0061]-page 21, lines 2-8, FIG. 7 - 704 |
| transmitting the self-expanding data package to a second computer system that expands the self-expanding data package into an expanded table having expanded table rows, | [0065]-page 21, line 24-page 22, line 5 and [0067]-page 22, lines 16-20; FIG. 7-708 |
| wherein each expanded table row represents a product and comprises a combination and each combination is generated by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values, | [0067]-page 22, lines 16-20; FIG. 7 - 708; [0006]-page 3, lines 14-20, [0007]-page 3, line 21-page 4, line 3, [0026]-page 7, line 22-page 8, line 4, [0057]-page 19, line 23-page 20, line 4, and [0068]-page 22, line 21-page 23, line 2; |
| wherein the one or more calculations eliminate one more expanded table rows. | [0007]-page 3, line 21-page 4, line 3, [0064]-page 21, lines 19-22, [0066]-page 22, lines 6-15, and [0068]-page 22, line 21-page 23, line 2; FIG. 7 - 708 |
|  |  |
| 38. An apparatus for generating product data in a self-expanding data package in a computer system comprising: | [0006]-page 3, lines 14-20, FIG. 7; [0022]- page 6, line 20-page 7, line 2. |
| (a) a computer system having a memory and a data storage device coupled thereto; | [0018]-[0021]-Page 5, line 16-page 6, line 19; FIG. 2 -204 and 214. |
| (b) one or more computer programs, performed by the computer system, for generating a self-expanding data package and storing the self-expanding data package in the | [0018]-[0021]-Page 5, line 16-page 6, line 19; FIG. 2 -204 and 214; [0030]-[0032]- page 8, line 20-page 9, line 14, [0036]-page 10, lines 11-15, and [0061]-page 21, lines 2-8, FIG. 7 - 700, 702, |

| | |
|---|---|
| memory, wherein the self-expanding data package is for product data and comprises: | and 708; [0026]-page 7, line 22-page 8, line 4, [0057]-page 19, line 23-page 20, line 4, [0067]-page 22, lines 16-20, and [0068]-page 22, line 21-page 23, line 2 |
| (i) one or more values in a set of one or more constant lists; and | [0030]-[0032]- page 8, line 20-page 9, line 14, [0036]-page 10, lines 11-15, and [0061]-page 21, lines 2-8, FIG. 7 - 700, 702, and 708; [0026]-page 7, line 22-page 8, line 4, [0057]-page 19, line 23-page 20, line 4, [0067]-page 22, lines 16-20, and [0068]-page 22, line 21-page 23, line 2 |
| (ii) one or more calculations that operate on one or more values in the set of one or more constant lists; | [0030]-[0032]- page 8, line 20-page 9, line 14, [0037]-page 10, line 18-page 11, line 4, and [0061]-page 21, lines 2-8, FIG. 7 - 704 |
| wherein the self-expanding data package is transmitted to a second computer system that expands the self-expanding data package into an expanded table having expanded table rows, | [0065]-page 21, line 24-page 22, line 5 and [0067]-page 22, lines 16-20; FIG. 7-708 |
| wherein each expanded table row represents a product and comprises a combination and each combination is generated by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values, | [0067]-page 22, lines 16-20; FIG. 7 - 708; [0006]-page 3, lines 14-20, [0007]-page 3, line 21-page 4, line 3, [0026]-page 7, line 22-page 8, line 4, [0057]-page 19, line 23-page 20, line 4, and [0068]-page 22, line 21-page 23, line 2; |
| wherein the one or more calculations eliminate one more expanded table rows. | [0007]-page 3, line 21-page 4, line 3, [0064]-page 21, lines 19-22, [0066]-page 22, lines 6-15, and [0068]-page 22, line 21-page 23, line 2; FIG. 7 - 708 |
| | |

| | |
|---|---|
| 64. An article of manufacture comprising a program storage medium readable by a computer and embodying one or more instructions executable by the computer to perform a method for generating product data in a self-expanding data package in a computer system, the method comprising: | [0006]-page 3, lines 14-20, FIG. 7; [0022]- page 6, line 20-page 7, line 2. |
| generating, in the self-expanding data package, one or more values in a set of one or more constant lists, wherein the self-expanding data package is for product data; | [0030]-[0032]- page 8, line 20-page 9, line 14, [0036]-page 10, lines 11-15, and [0061]-page 21, lines 2-8, FIG. 7 - 700, 702, and 708; [0026]-page 7, line 22-page 8, line 4, [0057]-page 19, line 23-page 20, line 4, [0067]-page 22, lines 16-20, and [0068]-page 22, line 21-page 23, line 2 |
| generating, in the self-expanding data package, one or more calculations that operate on one or more values in the set of one or more constant lists; | [0030]-[0032]- page 8, line 20-page 9, line 14, [0037]-page 10, line 18-page 11, line 4, and [0061]-page 21, lines 2-8, FIG. 7 - 704 |
| wherein the self-expanding data package is transmitted to a second computer system that expands the self-expanding data package into an expanded table having expanded table rows, | [0065]-page 21, line 24-page 22, line 5 and [0067]-page 22, lines 16-20; FIG. 7-708 |
| wherein each expanded table row represents a product and comprises a combination and each combination is generated by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values, | [0067]-page 22, lines 16-20; FIG. 7 - 708; [0006]-page 3, lines 14-20, [0007]-page 3, line 21-page 4, line 3, [0026]-page 7, line 22-page 8, line 4, [0057]-page 19, line 23-page 20, line 4, and [0068]-page 22, line 21-page 23, line 2; |
| wherein the one or more calculations eliminate | [0007]-page 3, line 21-page 4, line 3, [0064]-page |

| | |
|---|---|
| one more expanded table rows. | 21, lines 19-22, [0066]-page 22, lines 6-15, and [0068]-page 22, line 21-page 23, line 2; FIG. 7 - 708 |

## VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether claims 12-16, 18-24, 38-42, 44-50, 64-68, and 70-76 are unpatentable under 35 U.S.C. § 102(e) as being anticipated by U.S. Publication No. 2002/0107761 to Kark (Kark).

Whether claims 17, 43, and 69 are unpatentable under 35 U.S.C. §103(a) as being unpatentable over Kark.

## VII. ARGUMENTS

### A. Claims 12-16, 18-24, 38-42, 44-50, 64-68, and 70-76 are Patentable under 35 U.S.C. §102(e) over Kark

#### *1. Independent Claims 12, 38, and 64 Are Patentable Over The Prior Art*

Applicants traverse the rejections set forth in the final Office Action for one or more of the following reasons:

(1) Kark fails to teach, disclose or suggest a self-expanding data package;

(2) Kark fails to teach, disclose or suggest single package that contains both a set of constant lists and calculations that are performed on combinations of the constant lists;

(3) Kark fails to teach, disclose or suggest such calculations that eliminate rows/combinations of such constant lists; and

(4) Kark fails to teach, disclose or suggest transmitting a package that has constant lists and calculations to a second computer where it is expanded into a table.

Applicants note that Kark is directed to improving channel sales support in electronic commerce. In Kark, catalogs are maintained at different levels (see Kark [0039]). New catalogs can be created for particular users (see [0052]). However, the overriding principle of Kark lies in that a master catalog exists in a database and subsets of the master catalog can be created for users (see [0038]). However, what is clearly lacking from Kark is any concept of a self-expanding data package. In this regard, there is a complete lack of even the remote capability to transmit a single package that contains both values in constant lists and calculations. Further, Kark lacks the ability for a second computer to receive and expand the self-expanding package into an expanded table having rows based on combinations of the values in the different constant lists and calculations that eliminate some of the rows. Nowhere in Kark is such a possibility or teaching even remotely hinted at. Accordingly, not only has the Examiner failed to establish a prima facie case of lack of obviousness, but there isn't even a remote possibility for Kark to anticipate the present invention under the novelty standard of 35 U.S.C. §102.

Further details of the rejection and the lack of teaching are set forth below.

In rejecting the claims, the Office Action first asserts that paragraphs 0085 and 0092-0095 teach the limitation of generating the values in a set of one or more constant lists and storing the values in the self expanding data package. Paragraph [0085] provides:

> [0085] A catalog typically includes a plurality of products. The defined products are entered as records in the PRODUCT table 504. As above, the NAME field provides an owner defined name for the product defined by each record. The PRODUCTID is a unique value used as a key for the table and the CATALOGID links each record to the particular catalog in which it is included. The LINKFROM field in this table indicates another product, if any, from which this record represents a sub-product or sub-component. Such linking of products permits recursive definition of the products to provide a richer definition of products and related sub-products. For example, "TABLE SAW" may be the name of a product and "SAW BLADE" may be the name of a related sub-product. The LINKFROM field of a highest level product (one which is not a sub-product of another product) indicates the CATEGORYID value of the category, if any, to which it is linked, or, if not included in a category, the LINKFROM field for such a product indicates a NULL or other invalid value to so indicate a top level product.

Paragraphs [0092]-[0095] provide:

> [0092] Other tables shown in FIG. 5 further enhance the functionality and richness of the catalog product information. In particular, both products and categories may have attributes associated therewith. Examples of attributes for a product might be color or size. Attributes may be similarly applied to a category. For example, a clothing category might be "FABRIC" and the value of that attribute might be "COTTON." Such an attribute applied to a category implies all products within that category share that same attribute unless an individual product overrides that attribute value. A clothing product may similarly have such a FABRIC attribute and attribute value of COTTON. A product might also have a different FABRIC attribute value though it also is shown to be included in a category with the COTTON attribute value. In such a case, the product attribute value overrides the corresponding attribute value that may be associated with a category in which the product is included.
>
> [0093] The ATTR table 510 contains records with attributes defined for a related product or category. An ATTRID field provides a unique ID value for each attribute record and is used as a key in the table. The CATALOGID field, as above, is a key in the table and relates all records in the table to a particular catalog. A LINKFROM field indicates the PRODUCTID or CATEGORYID to which the attribute record is related. A NAME field provides a user-defined (owner-defined) name to the attribute. The VISIBILITY field is a test vehicle as described above with respect to other tables. The CATALOGID and LINKFROM fields may be used as indices on the table. ISREQUIRED, ISKEY

> and STYLE are fields that help define the usage of the attribute, i.e., whether it is required, etc.
>
> [0094] Attribute records in the table have one or more values associated therewith. If the values are simply defined, the HASVAL field of the record indicates that the VALUETYPE and VALUE fields of the record define the possible values. The VALUETYPE field indicates whether the attribute has a single value, a list of enumerated values, or a more complex set of possible ranges or domains of possible values. If VALUETYPE indicates that the attribute record includes a single possible value for the attribute, then the permitted value is stored in the VALUE field. If VALUETYPE indicates that the attribute has a list of possible values, entries in the VALUES table 516 are linked to the attribute record to specify the list of possible values. Specifically, the LINKFROM field of the VALUES table 516 entries corresponding to the attribute record have the ATTRID value to indicate the linkage. As with other tables discussed above, the VALUES table 516 entries include a CATALOGID field as a key value to associate the records with a particular catalog. The VISIBILITY field of the VALUES table 516 entries is a test vehicle as described above. The VALUE field of each VALUES table 516 entry has one of the list of possible values of the related attribute record.
>
> [0095] The ATTR table 510 records may alternatively indicate that the record has a more complex range of values that are permitted. The HASVALDOMAIN field of the attribute record indicates that records in the VALUEDOMAIN table 514 define the range(s) of permissible values of the attribute record. The VALUEDOMAIN table 514 records include a CATALOGID field as a key to associate the records of the table with a particular catalog and a VISIBILITY field as defined above. The LINKFROM field of the VALUEDOMAIN table 514 records link back to the ATTRID value of the related attribute record. The SELECTION, INTERVAL and OPT fields of the VALUEDOMAIN table 514 records define the complex range of attribute values permissible for the related attributes record.

As can clearly be seen, nowhere in such text is there even a remote capability for a self-expanding data package. In this regard, nothing in the text or remainder of Kark is anything expanded in any way, shape, or form. Further, the ability to have a self-expanding package is not hinted at, alluded to, or anticipated whatsoever. Instead, paragraph [0085] describes a product table having various fields and link fields that point to other products related to the current product. Such a link merely refers to other products and does not provide for a package whatsoever. Instead, it is merely a table. Similarly paragraphs [0092]-[0095] provide for various attributes that may exist for a product. Records in a table may be related to a particular catalog. In addition, link fields point to possible values for a given field. However, nowhere is such information put into a package as claimed. Further, nowhere is such information placed into a data package that is self expanding as claimed.

The next claim element is that of generating calculations that are placed in the same self-expanding data package. To teach this claim element, the Office Action relies on paragraphs [0052], [0055], [0092]-[0095], and [0098]. However, such paragraphs do not provide for any such calculations. Instead, paragraph [0052] merely provides the ability to filter catalog information that is presented to a user through the use of  a "stoplist data structure". Firstly, such a stoplist is not equivalent to calculations that operate on values in the constant lists. Secondly, Kark's stoplist is not stored in the same package as that of the values referred to in the prior paragraph. Instead, the stoplist is merely associated with the database (see paragraph [0055]). In fact, paragraph [0098] which is relied upon by the Examiner explicitly provides that the stoplist is a table that is defined in a database and is not necessarily associated with a particular catalog. A user accessing a catalog would invoke inspection of a particular stoplist table to filter products not intended to be viewed. Thus, rather than teaching the storage of the stoplist in the same package as the values (as presently claimed), Kark explicitly teaches away from such storage by stating that the stoplist is a separate table stored in a database and the table is accessed by one desiring to view a catalog. Thus, Kark teaches away from this claim limitation.

The next claim element provides for transmitting the self-expanding package to a second computer that expands the package into a table with each row as a combination of every value in each constant list and calculations performed on the values and the calculations eliminate one or more rows of the table. In rejecting this claim element, the Office Action merely relies on paragraphs [0038], [0052], [0055], [0092]-[0095], [0098], and [0105]. Upon examination of such paragraphs it may clearly be seen that Kark does not even remotely suggest such a claim limitation nor the claim as a whole.

Paragraph [0038] is the summary of the invention and describes how there are different levels of a catalog with each having a customized catalog version. A master catalog is also provided. Paragraph [0052], [0055], [0092]-[0095], and [0098] are described above. Paragraph [0105] describes the ability to duplicate a catalog's contents into a new catalog wherein all of the content of a parent database to a new child database followed by copying all of the fields of all tables.

As can be seen, such paragraphs do not provide for the ability to expand a package that contains specific information into a particular formatted table as claimed. Instead, Kark merely describes the ability to duplicate tables using various databases. Nowhere is there even a remote concept of a second computer receiving a self-expanding package that contains values in constant lists and calculations, where the second computer then expands the package into a table by combining all of the values and performing calculations to eliminate certain rows of the table. One unique advantage of such an invention lies in the ability to send a compact package with just the values and a set of calculations over a connection where it is then expanded. Such an advantage is reflected in the title of the invention "Lightweight Self-Contained Self-Expanding Product Data Package". The presently claimed limitations serve to provide such advantages. However, Kark does not even recognize the problem of sending large catalogs of information and instead actually teaches the use of multiple full catalogs - thereby remaining consistent with all of the prior art described in the present patent application.

Further, Kark actually teaches away from the claimed self-expanding data package and the claimed expansion process. Thus, Kark fails to even remotely suggest multiple aspects of the presently claimed invention. Accordingly, it is impossible for Kark to anticipate the present invention under the novelty standards of 35 U.S.C. §102.

Again, the presently claimed invention sets forth and establishes specific limitations for what is included in the self-expanding data package as well as how the package is expanded and used. Kark completely fails to teach, disclose, or suggest both claim aspects. Kark further fails to recognize the problem and the solution of the presently claimed invention.

In response to the above, the Examiner's answer initially asserts:

> …it is first noted that "self-expanding data package" does not implicitly hold the definition of "containing both values in constant lists and calculations". For instance, a self extracting Zip file might be considered a "self-extracting data package".

Appellants respectfully disagree with and traverse such an assertion. Firstly, not only do the claims recite "a self-expanding data package", but the claims explicitly recite and require that the self expanding data package has the following stored in the package: (1) values in a set of one or more constant lists and (2) calculations that operate on the one or more values in the set of one or more constant lists. Thus, the claimed self-expanding data package is explicit. Accordingly, any "implicit" definition that the Patent Office is relying upon is improper and in error. In addition, the Patent Office is now reciting a self extracting Zip file without any evidentiary support, explanation, or analysis of how such a Zip file meets the express claim limitations. Such a reliance is again in error and fails to establish a prima facie case of anticipation or obviousness.

The Answer continues on page 10 and asserts that Kark discloses a self-expanding data package wit the values in constant lists and calculations based on Kark paragraph 0085 with products entered as records in a table which the Answer asserts is "clearly a list". The Answer further asserts that paragraph 0052's stop list reads on the claimed calculations "because a stoplist would require a calculation (i.e., if a product has a particular PRODUCTID then exclude the product from view by the user)."

Appellants direct the attention of the Board to the arguments above. Again, paragraph [0085] describes a product table having various fields and link fields that point to other products related to the current product. Such a link merely refers to other products and does not provide for a package whatsoever. Instead, it is merely a table. In addition, as described above, the stop list is a separate table stored in a database and the table is accessed by one desiring to view a catalog. Further, filtering a list of products to be viewed based on a separate table is not even remotely similar to calculations that operate on values in a set of constant lists. In addition, the claims explicitly require that both the calculations and values are stored in the same self-expanding data package. Kark teaches away from such a limitation since Kark explicitly provides for using a separate table for the stoplist. Again, there is not even a remote contemplation of a package or a self-expanding package in any way, shape, or form in Kark.

On pages 10-11, the Answer further asserts:

> There are many parts of Kark that could be considered the "self-expanding" part of the data package. This includes the fact that Kark discloses a product hierarchy including "LINKFROM" field which represent sub-products and sub-subcomponents in order to provide a richer definition of products and their related sub-products. See paragraph 0085. Further as disclosed in paragraph 0094, attribute records in the product table can have one or more values associated therewith. The VALUETYPE fields indicates whether the attribute has single value, a list of enumerated values, or a more complex set of possible ranges or domains of possible values". If there are multiple values for a particular attribute the VALUES table is used to convey all the possible choices for that attribute for that product. The product table is then expanded, again with a "LINKFROM" field.

Appellants respectfully disagree with and traverse such assertions. The self-expanding part of the claims explicitly provide for creating an expanded table having expanded table rows, where each expanded table row represents a product and comprises a combination and each combination is generated by combining every value in each constant list with any combination of values from remaining parameters and performing the calculations on the values, where the calculations eliminate one or more expanded table rows. The Examiner is now asserting that a LINKFROM field teaches such a self-expansion. Not only does the LINKFROM field not expand anything, but there is no table that is expanded, the LINKFROM field does not combine every value in each constant list with any combination of values, the LINKFROM field does not perform any calculation on a value and the LINKFROM field not eliminate an expanded table row. In addition, the LINKFROM field is not in a package at all. Instead, the LINKFROM field is merely a reference to other products.

The Examiner relies on paragraph 0094 and asserts that a VALUES table is expanded using the VALUETYPE field and further expanded with the LINKFROM field. As set forth in paragraph 0094, the VALUE field of a record define possible values and the VALUETYPE field indicates whether the attribute has a single value, a list of enumerated values, or a more complex set of possible ranges or domains of possible values. If VALUETYPE indicates a single value, the single value is stored in the VALUE field of the VALUES table. If VALUETYPE indicates a list of possible values, entries in the VALUES table are linked (via the LINKFROM field) to the attribute record to specify the list of values. However, what is clearly and notoriously lacking from such a VALUES table is any expansion of more rows whatsoever. Instead, values are merely set in the existing entries or alternatively, a LINKFROM field is used to link to additional values for a particular field. Contrary to that asserted in the Answer, the VALUES table is not expanded whatsoever. Further, the VALUES tables does not have expanded table rows. Further yet, the VALUES table (and more specifically expanded table rows of the VALUES table) is not produced by combining every value in each constant list with any combination of values from parameters and performing calculations (all of which are stored in the package itself). There is simply no comparison of Kark's VALUES table to the present invention. Further, the LINKFROM field does not create expanded table rows whatsoever.

Page 11 of the Answer asserts that paragraph 0077 teaches an object-oriented database is equivalent to the claimed packaged structure and paragraph 0105 discloses the copying of a catalog to create a new catalog while figure 4 and paragraph 0075 disclose different versions of the database being located at different locations after being transferred. While copying an entire database may be described in Kark, such a copying is not what is claimed. In this regard, the Answer is not focusing on the claim language but on what Kark teaches and not the applicability of such a teaching to the claims and is therefore in error and fails to establish a prima facie case of unpatentability. Namely, the claims provide for transmitting a self expanding data package that in turn expands the package in a certain manner. Further, the claims explicitly provide that the package contains values in constant lists and calculations that operate on the values. Neither Kark's database nor Kark's catalog teach a single package that has both the values and the calculations as claimed. In this regard, a database is not self-expanding. Instead, it is merely a database that contains tables.

Further, the problem and solution of the present invention are not even remotely addressed by Kark. As stated in the present specification, one problem with the prior art is the size of a product catalog and the transmission of such a catalog. The present invention solves this problem using the self-expanding data package which can be easily transmitted and then expanded on the receiving side. Nothing in Kark even remotely alludes to such a solution or the possibility of using Kark to solve the problem presented by the invention.

On page 11, the Answer continues and asserts that to meet all of the claim limitations, Kark need only disclose one value in one list that has one calculation to be performed on it. Appellants respectfully disagree. Again, the claims provide for (1) generating one or more values in a set of one or more constant lists and (2) generating one or more calculations that operation on the one or more values. What the Answer is ignoring is that claims also provide that the package is expanded into an expanded table having expanded table rows with each row as a combination that is generated by combining every value in each constant list with any combination of values and performing the calculations no the values. The claims further explicitly provide that the calculations eliminate one or more expanded table row. Thus, merely having one value and one calculation completely fails to account for the remaining the elements that utilize the value and calculation to produce expanded table rows and eliminate one or more such rows. Kark fails to teach, disclose, or suggest such elements, explicitly and implicitly.

The Answer continues at the bottom of page 11 and while relying on paragraph 0077 and 0105 asserts that Kark placing tables in a relational or obejct-oriented database and copying the database to create a new version of the catalog on another server teaches the claimed self-expanding data package. For the reasons set forth above, Appellants respectfully disagree with and traverse such assertions. Again, neither Kark's database nor his catalog is even remotely similar to the claimed self-expanding package and the ability to have values and calculations where table rows are expanded and eliminated based on calculations. None of the claimed capabilities are taught, suggested, or alluded to in Kark.

The Answer then states that Kark's database is "self-expanding" because of fields liked to other tables. Again, the self expanding components of the claims provide not only for a single package to contain the constant lists (which have values) and the calculations but such data is used to create an expanded table having expanded table rows by combining particular elements with each other in a particular manner. Kark's LINKFROM field merely points to locations and does not expand anything nor does it expand table rows or eliminate table rows whatsoever.

The Answer asserts on page 12 that Kark's links expand a simple table to include other values that are not found in the table but are found in other tables and therefore expands the information found in the original table. Again, the claims do not provide for merely expanding a table but provide for expanded table rows with each row represented and product and comprising a combination. Further, each combination is generated not by following links to other tables with listed attributes as in Kark, but by combining every value in each constant list with any combination of values from remaining parameters. Further, the calculations that are also in the package are used to eliminate one or more expanded table rows. Kark's tables are not expanded in any manner that is even remotely similar to such a claimed expansion. Instead, the database is transmitted in its entirety. Such a database transmission would serve to teach away from the present invention because such a database transmission would not solve the problems of the prior art and would further require multiple tables with explicit references to each other that are not combined together.

On page 12, the Answer asserts that the stoplist data structure teaches the calculations because if a certain situation occurs, items are not displayed and are removed from the client view. Appellants note that the STOPLIST is not equivalent to the claimed calculations. First, the claimed calculations serve to eliminate actual expanded table rows from an actual table. However, Kark's stoplist merely provide for a portal that is presented to the user. Kark's entire database still exists. In fact, Kark explicitly states (in paragraph [0098]):

> A user accessing the catalog, for example, by a particular portal would invoke inspection of a corresponding STOPLIST table 520 record, if any, to filter products not intended for presentation to that user over that particular portal. In the preferred embodiment, all catalogs physically reside at a service provider's site such that all access is centralized and controlled by a catalog access service. The service determines through means outside of the present invention what particular portal is used to access the database and then looks for records in the STOPLIST table 520 that indicate the presence

> of items not to be presented to users of the catalog accessing the catalogs through the corresponding portal. In this preferred embodiment, the STOPLIST table 520 and associated items in the STOPLIST_ITEM table 522 are defined by the catalog service provider in conjunction with the portal management.

As can clearly be seen, Kark's STOPLIST does not eliminate anything from an actual expanded table that has been created. Instead, it merely presents a focused portal or view of the entire catalog. As further stated in paragraph [0098], Kark's STOPLIST is used in e-commerce marketing arrangements that users accessing a catalog from a particular portal should see a preference for a particular brand or brands of goods to the exclusion of other goods. Again, no rows of a table are eliminated from a table.

The Answer then asserts that the copying of a catalog as a starting point for a new product information catalog as shown in Kark's FIG. 4 and paragraph [0075] teaches the various claim elements. Appellants again note that the method that is used to expand the table and the contents of the package that are transmitted (and then used to expand the table and table rows) is neither taught not suggest by the copying of a catalog. Kark's FIG. 4 merely shows a catalog structure of an e-commerce marketing channel with multiple catalogs being provided in their entirety and integrated into a master catalog. There is no combination as claimed nor is there an expansion of table rows as claimed. Instead, Kark teaches away from such a single self-expanding package by requiring multiple different catalogs that are then integrated together into a master catalog without combining values in constant lists with values from remaining parameters followed by calculations that eliminate rows from the table (as claimed). In this regard, Kark is not even remotely similar to the claimed invention.

The remaining arguments on pages 13-14 of the Answer also fail to teach the invention as claimed. Again, Kark is directed to a different problem, proposes a different solution, and fails to teach numerous claim elements.

In view of the above, Appellants respectfully request submit that not only is the Office Action and Answer in error but the Action further fails to establish a prima facie case of unpatentability. Accordingly, Appellants respectfully request reversal of the rejections.

### *2. Dependent Claims 13, 39, and 65 Are Patentable Over the Cited Art*

Dependent claims 13, 39, and 65 provide for storing a basic table in the same self-expanding data package. Further, the expanded table is further expanded by combining every value in each constant list with each row in the basic table.

In rejecting these claims, the Office Action relies on paragraphs [0085] and [0092]-[0095]. Appellants note that as stated above, nowhere does Kark even remotely allude to a package whatsoever that contains the values and calculations. Further, the ability to store a table in the same self expanding package that is then used to create an expanded table is completely absent from Kark. Again, the claims provide for storing multiple items in a single package that is then expanded by a second computer into another table. Nowhere is such a concept remotely hinted at anywhere in Kark.

### *3. Dependent Claims 14, 40, and 66 Are Patentable Over the Cited Art*

Dependent claims 14, 40, and 66 provide that calculations are applied to test the validity of the expanded table rows and only valid rows are maintained in the table.

The Office Action rejects such claim limitations based on paragraphs 0052 and 0098. Again, paragraph 0052 merely describes that a catalog can be filtered using a stoplist data structure. Paragraph 0098 describes the use of the stoplist table. However, nowhere is there any mention in either paragraph of whether or how a row can be valid or not. Further, nowhere is there any test of validity of such a row in a stoplist. Instead, a stoplist is merely a table that defines products that are not intended to be presented to users depending on factors such as the portal used to access the catalog. Whether a row item is valid or the ability to test the validity of a row item is entirely lacking from Kark.

In response to the above arguments, the Answer relies on Kark paragraph 0055 which defines the stoplist as identifying products in corresponding catalogs to be excluded from presentation to a requesting user based on portal used for accessing the database by the user. The Answer then asserts that a stoplist removing all items that are not valid to the portal the user is using. Appellants respectfully disagree. Removing a particular product from a view that is presented to a user does not reflect on the validity of a row of a table whatsoever. The claims require that only those expanded table rows that are valid are maintained in the expanded table. Kark does not remove anything from any table. Instead, all items are maintained in Kark's table and a restricted view is presented to the user. Again, nothing regarding the validity or maintaining particular valid rows in a table are even hinted at in Kark.

In view of the above, Appellants respectfully request reversal of the rejection of claims 14, 40, and 66.

### 4. *Dependent Claims 15, 41, and 67 are Patentable Over the Cited Art*

Dependent claims 15, 41, and 67 provide that the calculations of claims 14, 40, and 66 are used to perform a precursor conditional test that is used to test the validity of the expanded table rows.

In rejecting these claims, the Action again relies on paragraphs 0052 and 0098. However, such text does not describe or allude to a precursor conditional test that is performed on a row of a table. Again, the validity of a row and the ability to perform any test, precursor or otherwise is nowhere to be found in Kark.

The Answer asserts that Kark's products are removed from the product table before being displayed to the user. Such an assertion is meritless and contrary to that direct teaching of Kark as described above. In this regard, Kark does not delete anything from the tables but explicitly maintains the tables and merely presents a filtered view to a particular portal. Accordingly, the Office Action and Answer are mischaracterizing the reference, and are therefore in error and fail to establish a prima facie case of unpatentability.

In view of the above, Appellants respectfully request reversal of the rejection of claims 15, 41, and 67.

### 5. *Dependent Claims 16, 42, and 68 are Not Separately Argued*

### 6. *Dependent Claims 18, 44 and 70 are Patentable Over the Cited Art*

Claims 18, 44, and 70 provide that the calculations eliminate duplicate expanded table rows.

In rejecting these claims, the final Office Action merely refers to paragraph [0107] which provides:

> [0107] FIG. 25 is a flowchart describing a manual process for updating information in a catalog from its parent catalog. Those skilled in the art will recognize, as noted above, that such update processing may be automated on a periodic basis rather than performed as shown here as a manual process. Such automation and periodic operation may be implemented using well-known scripting and task scheduling features common in most computer operating systems.

As can be clearly seen, nowhere is there even a remote reference in such text to eliminating a duplicate row whatsoever. Such text merely describes a manual process for updating information in a catalog. Fig. 25 also fails to teach, describe, or suggest any form of eliminating duplicate table rows. Appellants further submit that since Kark's catalogs are not created using values and constants as presently claimed, duplicate rows would not exist in Kark and therefore, there would be no need to eliminate them as set forth in these claims.

In response to the above, the Answer acknowledges paragraph [0107] was recited in error and now relies on paragraph 0082 which teaches having different versions of the same catalog. The Answer further asserts that a product having more than one version would have duplicate items and displaying a particular version to the end user would eliminate the duplicate items. Paragraph 0082 provides:

> [0082] The CATALOG table 500 contains a record for each catalog defined by the owner of the enterprise. In addition to the CATALOGID field, a VERSION and NAME field help identify the particular catalog in accordance with the owner's convention. For example, the catalog may be the "Clothing" catalog by name and the "Fall 2000" catalog by version. Though the NAME field may preferably be used as a common index on the table, it is not assured to be unique and therefore is not the preferred key for the table. The PARENTCATALOGID contains the CATALOGID value of the parent catalog from which this catalog was derived. As used herein, a "parent" catalog refers to a catalog at the next higher layer of the marketing hierarchy of catalogs and a "child" catalog refers to a catalog at the next lower layer of the marketing catalog hierarchy.

As can be seen, paragraph [0082] describes different catalogs with different versions that may for example be used for different purposes (e.g., a Fall clothing catalog v. different clothing catalogs). The Examiner is asserting that there are different versions of the same catalog and that there is likely to be duplicate items. Such a theory is merely a theory and not based on any teaching in the cited reference. The rejection is based on a standard of anticipation and not obviousness and Kark does not teach any duplicate versions of the same catalog in paragraph [0082].

Further, assuming (for purposes of the argument) that different versions are disclosed, the Examiner makes an additional assumption in that one version would have duplicate items. The example in paragraph [0082] provides for a Fall 2000 catalog. To assume that a fall 2000 catalog would have the same clothing as a Spring 2000 catalog is illogical in that most designers have different lines from year to year and season to season.

Based on the above, it can clearly be seen that the rejection is in error and fails to establish a prima facie case of unpatentability.

However, in addition to the above, the Examiner again asserts that displaying a particular version would eliminate duplicate items. The claims do NOT provide for only displaying a particular version of a catalog. Instead, the claims explicitly provide for eliminating particular expanded table rows from an existing table. The showing of a particular version of a catalog which doesn't have products from another catalog does not even remotely allude to a calculation that actually eliminates a row from a table. Again, only displaying a particular version of a catalog does not perform a calculation whatsoever.

In view of the above, Appellants respectfully request reversal of the rejection of claims 18, 44, and 70.

*7. Dependent Claims 19, 45 and 71 are Not Separately Argued*

*8. Dependent Claims 20, 46 and 72 are Not Separately Argued*

*9. Dependent Claims 21, 47 and 73 Are Not Separate Argued*

*10. Dependent Claims 22, 48 and 74 Are Not Separate Argued*

*11. Dependent Claims 23, 49 and 75 Are Not Separately Argued*

*12. Dependent Claims 24, 50 and 76 Are Patentable Over the Cited Art*

Dependent claims 24, 50, and 76 provide that the logic for expanding the data into the expanded table is fully defined within the data package and the data.

In rejecting these claims, the final Office Action merely refers to Kark paragraphs 0052, 0055, 0092-0095, and 0098. Appellants note that the logic for actually expanding the data must be contained and defined in the package (as claimed). Kark fails to provide any sort of self-expanding package whatsoever. Further, nowhere in Kark is there any capability to place logic for expanding a package into the package itself. In fact, since Kark fails to even remotely allude to a package whatsoever, it would be impossible for Kark to include logic used to expand such a package into a table as claimed.

Appellants note that due to the total lack of similarity between Kark and the present invention, it appears difficult to recite any elements of Kark that even remotely refer to the purpose, subject matter, problem, or solution of the presently claimed invention. Instead, Kark describes a structure that is entirely different and based on different concepts than that of the presently claimed invention. Not only does Kark lack the logic used to expand the claimed package, but the placement of such logic in the package itself is nowhere to be found in Kark. Instead, Kark describes a database with different tables and links - such a description is not and cannot be equivalent to the claimed logic.

In response to the above, the Answer asserts that Kark's database is defined with schema and built into schema is relationships between the tables which are used to build the product view given to the client. The Answer further asserts that part of the defined schema entails a stoplist that eliminates some of the items that can be viewed by a client which teaches a logic operation to remove those items.

Firstly, these claims provide that the logic for <u>expanding</u> the data package is included and fully defined within the data package and data. As described above, Kark's stoplist removes items from view and therefore on its face would not provide for the expansion of anything. Again, logic for expanding something cannot and is not equivalent to logic for eliminating something. Further, the stoplist is merely a table of records that include a unique value STOPLIST key field and VERSION, NAME, VISIBILITY, CREATE_DATE and CHANGE_DATE fields (see paragraph

[0098]). Such fields are not logic whatsoever. Instead, they are merely values that would be used to change a view exposed to a user in a particular portal.

In addition to the above, Appellants note that there is nothing in Kark that describes the incorporation (into the database) of any logic regarding how to combine any tables in Kark in any manner whatsoever. Such logic is wholly and completely lacking from Kark.

Again, as stated above, without teaching the possibility of combining elements to produce expanded table rules as well as the use of a "package" that is self-expanding and is transmitted, Kark cannot possibly teach the use of logic to expand a package, a table in the package, or the method used for expanding the package as expressly recited in the claims.

In view of the above, Appellants respectfully request reversal of the rejection of claims 24, 50, and 76.

## B. Claims 17, 43, and 69 are Not Separately Argued

## VIII. CONCLUSION

In light of the above arguments, Appellants respectfully submit that the cited references do not anticipate nor render obvious the claimed invention. More specifically, Appellants' claims recite novel physical features which patentably distinguish over any and all references under 35 U.S.C. §§ 102 and 103. As a result, a decision by the Board of Patent Appeals and Interferences reversing the Examiner and directing allowance of the pending claims in the subject application is respectfully solicited.

Respectfully submitted,

GATES & COOPER LLP
Attorneys for Applicant(s)

Howard Hughes Center
6701 Center Drive West, Suite 1050
Los Angeles, California 90045
(310) 641-8797

Date: December 5, 2008

By:/Jason S. Feldmar/
Name: Jason S. Feldmar
Reg. No.: 39,187

JSF/sjm

G&C 30566.203-US-01

## CLAIMS APPENDIX

1. (WITHDRAWN) A self-expanding product data package for product data comprising:

basic table data having one or more table rows, wherein each row represents a partial definition of a product;

a set of one or more constant lists having one or more values; and

one or more row validation calculations;

wherein use of the set of constant lists and row validation calculations provides a mechanism for compact data storage, wherein the self-expanding data package is capable of expansion into an expanded table having expanded table rows wherein each expanded table row represents a product and comprises a combination such that:

for each constant list of values, every list member is combined with all other basic table rows and additional list members to produce every possible combination; and

the row validation calculations are applied to test validity of the expanded table rows, and only those expanded table rows that are valid are maintained in the expanded table.

2. (WITHDRAWN) The self-expanding data package of claim 1 further comprising a calculation utilized to perform a precursor conditional test that is used in one or more row validation calculations.

3. (WITHDRAWN) The self-expanding data package of claim 1 further comprising a calculation utilized to provide additional data used in the expanded table

4. (WITHDRAWN) The self-expanding data package of claim 1 wherein the self-expanding data package comprises product data for use in a computer-aided design application.

5. (WITHDRAWN) The self-expanding data package of claim 1 wherein the one or more of the row validation calculations provide for eliminating duplicate expanded table rows.

6. (WITHDRAWN) The self-expanding data package of claim 1 wherein the basic table data, set of one or more constant lists, and one or more row validation calculations are specified using extensible markup language (XML).

7. (WITHDRAWN) The self-expanding data package of claim 1 wherein the row validation calculations are selected through a graphical user interface.

8. (WITHDRAWN) The self-expanding data package of claim 1 wherein the self-expanding data package is transmitted across a network.

9. (WITHDRAWN) The self-expanding data package of claim 1 wherein one or more row validation calculations comprise one or more filters that limit results displayed from the expanded table rows.

10. (WITHDRAWN) The self-expanding data package of claim 1 wherein an editor provides an ability to directly edit the self-expanding data package.

11. (WITHDRAWN) The self-expanding data package of claim 1 wherein logic for expanding the data package into the expanded table is fully defined within the data package and the data.

12. (PREVIOUSLY PRESENTED) A method for generating product data in a self-expanding data package in a computer system comprising:

generating one or more values in a set of one or more constant lists and storing said one or more values in the self-expanding data package, wherein the self-expanding data package is for product data;

generating one or more calculations that operate on one or more values in the set of one or more constant lists and storing said one or more calculations in the self-expanding data package;

transmitting the self-expanding data package to a second computer system that expands the self-expanding data package into an expanded table having expanded table rows, wherein each expanded table row represents a product and comprises a combination and each combination is generated by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values, wherein the one or more calculations eliminate one more expanded table rows.

13. (PREVIOUSLY PRESENTED) The method of claim 12 further comprising, generating one or more basic table data having one or more table rows, and storing said one or more basic table data in the self expanding data package, wherein the self-expanding data package is further expanded by combining every value in each constant list with each basic table row.

14. (PREVIOUSLY PRESENTED) The method of claim 12, wherein one or more calculations are applied to test validity of the expanded table rows, and only those expanded table rows that are valid are maintained in the expanded table.

15. (PREVIOUSLY PRESENTED) The method of claim 14, wherein one or more calculations are utilized to perform a precursor conditional test that is used to test validity of the expanded table rows.

16. (PREVIOUSLY PRESENTED) The method of claim 12, wherein one or more calculations are utilized to provide additional data used in the expanded table.

17. (ORIGINAL) The method of claim 12, wherein the self-expanding data package comprises product data for use in a computer-aided design application.

18. (ORIGINAL) The method of claim 12, wherein one or more calculations provide for eliminating duplicate expanded table rows.

19. (ORIGINAL) The method of claim 12, wherein the self-expanding data package is written in extensible markup language (XML).

20. (ORIGINAL) The method of claim 12, wherein one or more calculations are selected through a graphical user interface.

21. (ORIGINAL) The method of claim 12, wherein the self-expanding data package is transmitted across a network.

22. (ORIGINAL) The method of claim 12, wherein one or more calculations comprise one or more filters that limit results displayed from the expanded table rows.

23. (PREVIOUSLY PRESENTED) The method of claim 12, wherein an editor is used to directly edit the self-expanding data package.

24. (ORIGINAL) The method of claim 12, wherein logic for expanding the data package into the expanded table is fully defined within the data package and the data.

25. (WITHDRAWN) A method for utilizing product data in a self-expanding data package in a computer system comprising:

receiving a self-expanding data package comprising one or more values in a set of one or more constant lists and one or more calculations that operate on one or more values in the set of one or more constant lists, wherein the self-expanding data package is for product data;

expanding the self-expanding data package into an expanded table having expanded table rows, wherein each expanded table row represents a product and comprises a combination and each combination is generated by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values, wherein the one or more calculations eliminate one more expanded table rows.

26. (WITHDRAWN) The method of claim 25, wherein:

the self-expanding data package further comprises one or more basic table data having one or more table rows; and the expanding further comprises combining every value in each constant list with each basic table row.

27. (WITHDRAWN) The method of claim 25, wherein:

one or more calculations test validity of the expanded table rows; and

only those expanded table rows that are valid are maintained in the expanded table.

28. (WITHDRAWN) The method of claim 27, wherein one or more calculations perform a precursor conditional test that is used to test validity of the expanded table rows.

29. (WITHDRAWN) The method of claim 25, wherein one or more calculations provide additional data used in the expanded table

30. (WITHDRAWN) The method of claim 25, wherein the self-expanding data package comprises product data for use in a computer-aided design application.

31. (WITHDRAWN) The method of claim 25, wherein one or more calculations eliminate duplicate rows or otherwise apply business rules to eliminate unwanted rows in the resulting expanded table.

32. (WITHDRAWN) The method of claim 25, wherein the self-expanding data package is written in extensible markup language (XML).

33. (WITHDRAWN) The method of claim 25, wherein one or more calculations are selected through a graphical user interface.

34. (WITHDRAWN) The method of claim 25, wherein the self-expanding data package is received from across a network.

35. (WITHDRAWN) The method of claim 25, wherein one or more calculations comprise one or more filters that limit results displayed from the expanded table rows.

36. (WITHDRAWN) The method of claim 25, wherein an editor provides an ability to directly edit the self-expanding data package.

37. (WITHDRAWN) The method of claim 25, wherein logic for expanding the data package into the expanded table is fully defined within the data package and the data.

38. (PREVIOUSLY PRESENTED) An apparatus for generating product data in a self-expanding data package in a computer system comprising:

(a) a computer system having a memory and a data storage device coupled thereto;

(b) one or more computer programs, performed by the computer system, for generating a self-expanding data package and storing the self-expanding data package in the memory, wherein the self-expanding data package is for product data and comprises:

(i) one or more values in a set of one or more constant lists; and

(ii) one or more calculations that operate on one or more values in the set of one or more constant lists;

wherein the self-expanding data package is transmitted to a second computer system that expands the self-expanding data package into an expanded table having expanded table rows, wherein each expanded table row represents a product and comprises a combination and each combination is generated by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values, wherein the one or more calculations eliminate one more expanded table rows.

39. (PREVIOUSLY PRESENTED) The apparatus of claim 38, wherein:

the self-expanding data package further comprises one or more basic table data having one or more table rows; and

the self-expanding data package is further expanded by combining every value in each constant list with each basic table row.

40. (PREVIOUSLY PRESENTED) The apparatus of claim 38, wherein one or more calculations are applied to test validity of the expanded table rows, and only those expanded table rows that are valid are maintained in the expanded table.

41. (PREVIOUSLY PRESENTED) The apparatus of claim 40, wherein one or more calculations are utilized to perform a precursor conditional test that can be used to test validity of the expanded table rows.

42. (PREVIOUSLY PRESENTED) The apparatus of claim 38, wherein one or more calculations are utilized to provide additional data used in the expanded table

43. (ORIGINAL) The apparatus of claim 38, wherein the self-expanding data package comprises product data for use in a computer-aided design application.

44. (ORIGINAL) The apparatus of claim 38, wherein one or more calculations provide for eliminating duplicate expanded table rows.

45. (ORIGINAL) The apparatus of claim 38, wherein the self-expanding data package is written in extensible markup language (XML).

46. (ORIGINAL) The apparatus of claim 38, further comprising a graphical user interface displayed by the computer system for selecting the one or more calculations.

47. (ORIGINAL) The apparatus of claim 38, wherein the one or more computer programs are further configured to transmit the self-expanding data package across a network.

48. (ORIGINAL) The apparatus of claim 38, wherein one or more calculations comprise one or more filters that limit results displayed from the expanded table rows.

49. (PREVIOUSLY PRESENTED) The apparatus of claim 38, wherein one or more of the computer programs comprise an editor that is used to directly edit the self-expanding data package.

50. (ORIGINAL) The apparatus of claim 38, wherein logic for expanding the data package into the expanded table is fully defined within the data package and the data.

51. (WITHDRAWN) An apparatus for utilizing product data in a self-expanding data package in a computer system comprising:

(a) a computer system having a memory and a data storage device coupled thereto;

(b) one or more computer programs, performed by the computer system, for receiving a self-expanding data package stored in the memory, wherein the self-expanding data package is for product data and the self-expanding data package comprises :

(i) one or more values in a set of one or more constant lists; and

(ii) one or more calculations that operate on one or more values in the set of one or more constant lists; and

(c) one or more computer programs, performed by the computer system, for expanding the self-expanding data package into an expanded table having expanded table rows, wherein each expanded table row represents a product and comprises a combination and each combination is generated by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values, wherein the one or more calculations eliminate one more expanded table rows.

52. (WITHDRAWN) The apparatus of claim 51, wherein:

the self-expanding data package further comprises basic table data having one or more table rows; and

the computer program is further configured to expand the self-expanding data package by combining every value in each constant list with each basic table row.

53. (WITHDRAWN) The apparatus of claim 51, wherein one or more calculations are applied to test validity of the expanded table rows, and only those expanded table rows that are valid are maintained in the expanded table.

54. (WITHDRAWN) The apparatus of claim 53, wherein one or more calculations are utilized to perform a precursor conditional test that is used to test validity of the expanded table rows.

55. (WITHDRAWN) The apparatus of claim 51, wherein one or more calculations are utilized to provide additional data used in the expanded table

56. (WITHDRAWN) The apparatus of claim 51, wherein the self-expanding data package comprises product data for use in a computer-aided design application.

57. (WITHDRAWN) The apparatus of claim 51, wherein one or more calculations provide for eliminating duplicate expanded table rows.

58. (WITHDRAWN) The apparatus of claim 51, wherein the self-expanding data package is written in extensible markup language (XML).

59. (WITHDRAWN) The apparatus of claim 51, further comprising a graphical user interface displayed by the computer system for selecting the one or more calculations.

60. (WITHDRAWN) The apparatus of claim 51, wherein the self-expanding data package is received from across a network.

61. (WITHDRAWN) The apparatus of claim 51, wherein one or more calculations comprise one or more filters that limit results displayed from the expanded table rows.

62. (WITHDRAWN) The apparatus of claim 51, further comprising an editor performed by the computer system that provides an ability to directly edit the self-expanding data package.

63. (WITHDRAWN) The apparatus of claim 51, wherein logic for expanding the data package into the expanded table is fully defined within the data package and the data.

64. (PREVIOUSLY PRESENTED) An article of manufacture comprising a program storage medium readable by a computer and embodying one or more instructions executable by the computer to perform a method for generating product data in a self-expanding data package in a computer system, the method comprising:

generating, in the self-expanding data package, one or more values in a set of one or more constant lists, wherein the self-expanding data package is for product data;

generating, in the self-expanding data package, one or more calculations that operate on one or more values in the set of one or more constant lists;

wherein the self-expanding data package is transmitted to a second computer system that expands the self-expanding data package into an expanded table having expanded table rows, wherein each expanded table row represents a product and comprises a combination and each combination is generated by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values, wherein the one or more calculations eliminate one more expanded table rows.

65. (PREVIOUSLY PRESENTED) The article of manufacture of claim 64, the method further comprising, generating, in the self-expanding data package, basic table data having one or more table rows, wherein the self-expanding data package is further expanded by combining every value in each constant list with each basic table row.

66. (PREVIOUSLY PRESENTED) The article of manufacture of claim 64, wherein one or more calculations are applied to test a validity of the expanded table rows, and only those expanded table rows that are valid are maintained in the expanded table.

67. (PREVIOUSLY PRESENTED) The article of manufacture of claim 66, wherein one or more calculations are utilized to perform a precursor conditional test that is used to test validity of the expanded table rows.

68. (PREVIOUSLY PRESENTED) The article of manufacture of claim 64, wherein one or more calculations are utilized to provide additional data used in the expanded table

69. (ORIGINAL) The article of manufacture of claim 64, wherein the self-expanding data package comprises product data for use in a computer-aided design application.

70. (PREVIOUSLY PRESENTED) The article of manufacture of claim 64, wherein one or more calculations eliminate duplicate expanded table rows.

71. (ORIGINAL) The article of manufacture of claim 64, wherein the self-expanding data package is written in extensible markup language (XML).

72. (ORIGINAL) The article of manufacture of claim 64, wherein one or more calculations are selected through a graphical user interface.

73. (ORIGINAL) The article of manufacture of claim 64, wherein the method further comprises transmitting the self-expanding data package across a network.

74. (ORIGINAL) The article of manufacture of claim 64, wherein one or more calculations comprise one or more filters that limit results displayed from the expanded table rows.

75. (PREVIOUSLY PRESENTED) The article of manufacture of claim 64, wherein an editor is used to directly edit the self-expanding data package.

76. (ORIGINAL) The article of manufacture of claim 64, wherein logic for expanding the data package into the expanded table is fully defined within the data package and the data.

77. (WITHDRAWN) An article of manufacture comprising a program storage medium readable by a computer and embodying one or more instructions executable by the computer to perform a method for utilizing product data in a self-expanding data package in a computer system, the method comprising:

(a) receiving a self-expanding data package wherein the self-expanding data package is for product data and the self-expanding data package comprises:

(i) one or more values in a set of one or more constant lists; and

(ii) one or more calculations that operate on one or more values in the set of one or more constant lists; and

(b) expanding the self-expanding data package into an expanded table having expanded table rows, wherein each expanded table row represents a product and comprises a combination and each combination is generated by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values, wherein the one or more calculations eliminate one more expanded table rows.

78. (WITHDRAWN) The article of manufacture of claim 77, wherein:

the self-expanding data package further comprises basic table data having one or more table rows; and

the self-expanding data package is further expanded by combining every value in each constant list with each basic table row.

79. (WITHDRAWN) The article of manufacture of claim 77, wherein one or more calculations are applied to test validity of the expanded table rows, and only those expanded table rows that are valid are maintained in the expanded table.

80. (WITHDRAWN) The article of manufacture of claim 79, wherein one or more calculations are utilized to perform a precursor conditional test that is used to test validity of the expanded table rows.

81. (WITHDRAWN) The article of manufacture of claim 77, wherein one or more calculations are utilized to provide additional data used in the expanded table

82. (WITHDRAWN) The article of manufacture of claim 77, wherein the self-expanding data package comprises product data for use in a computer-aided design application.

83. (WITHDRAWN) The article of manufacture of claim 77, wherein one or more calculations provide for eliminating duplicate expanded table rows.

84. (WITHDRAWN) The article of manufacture of claim 77, wherein the self-expanding data package is written in extensible markup language (XML).

85. (WITHDRAWN) The article of manufacture of claim 77, wherein one or more calculations are selected through a graphical user interface.

86. (WITHDRAWN) The article of manufacture of claim 77, wherein the self-expanding data package is received from across a network.

87. (WITHDRAWN) The article of manufacture of claim 77, wherein one or more calculations comprise one or more filters that limit results displayed from the expanded table rows.

88. (WITHDRAWN) The article of manufacture of claim 77, wherein an editor provides an ability to directly edit the self-expanding data package.

89. (WITHDRAWN) The article of manufacture of claim 77, wherein logic for expanding the data package into the expanded table is fully defined within the data package and the data.

## EVIDENCE APPENDIX

None.

## RELATED PROCEEDINGS APPENDIX

None.